Journal of Information Assurance and Security 2 (2007) 275-287

Minimality Quality Criterion Evaluation for Integrated Schemas

Maria da Conceição Moraes Batista and Ana Carolina Salgado

Centro de Informática, Universidade Federal de Pernambuco,
Av. Professor Luis Freire s/n, Cidade Universitária, 50740-540 Recife – PE, Brasil
{mcmb, acs}@cin.ufpe.br

***Abstract*:** Integrated access to distributed data is an important problem faced in scientific and commercial applications. A data integration system provides a unified view for users to submit queries over multiple autonomous data sources. The queries are processed over a global schema that offers an integrated view of the data sources. Much work has been done on query processing and choosing plans under cost criteria. However, not so much is known about incorporating Information Quality analysis into data integration systems, particularly in the integrated schema. In this work we present an approach of Information Quality analysis of schemas in data integration environments. We discuss the evaluation of schema quality focusing in the minimality aspect and define some schema transformations to be applied in order to improve schema design.

***Keywords*:** Data Integration, Information Quality, Data Quality, Schema Quality, Minimality**.**

1. Introduction

Information Quality (IQ) has become a critical aspect in organizations and, consequently, in Information Systems research [11, 33, 40, 15, 16, 17]. IQ is a multidimensional aspect and it is based in a set of dimensions or criteria. The role of each one is to assess and measure a specific IQ aspect [32, 37, 40, 4, 3, 1, 41, 12, 13, 38, 42, 43].

The main feature of a data integration system is to free the user from knowing about specific data sources and interact with each one. Instead, the user submits queries to a *global* or *integrated schema*, which is a set of views, over a number of data sources, designed for a particular data integration application. Commonly, the tasks of query processing involving query submission, planning, decomposition and results integration are performed by a software module called *mediator* [44]. Each source publishes a data source schema with the representation of its contents. The mediator reformulates a user query into queries that refers directly to schemas on the sources. To successfully reformulate a query, the mediator uses a set of correspondences, called *schema mappings*. There are also the user schemas that represent the requirements of information defined for one user or a group of users.

To the best of our knowledge, IQ criteria concerning global schemas quality in data integration systems are not defined in literature. As earlier discussed in Kesh [18], we believe that an alternative to optimize query execution would be the construction of good schemas, with high quality scores, and we have based our approach in this affirmative.

In a data integration system, we consider the *schemas* as the structures exported by the data sources (source schemas), the structures that are used by users to build queries (users' schema) and the integrated schema.

As a starting point, we have compiled IQ classifications proposed in previous works [1, 14, 32, 33, 40, 41, 26, 27, 42] and adapted them to address schema quality in data integration systems. After this analysis we proposed some variations: some criteria are not considered (not applicable), and some were adapted to our environment. Consequently, we obtained a list of three IQ criteria presented in Table 1.

Table 1. IQ Criteria for schema quality analysis

IQ Criteria	Definition	Metrics
Schema Completeness	The extent to which entities and attributes of the application domain are represented in the schema	1 – (#incomplete items / #total items) [1]
Type Consistency	Data type uniformity across the schemas	1 – (#inconsistent schema elements / #total schema elements)[1]
Minimality	The extent in which the schema is modeled without redundancies	1 – (#redundant schema elements / #total schema elements)[1]

Schema Completeness

The completeness can be measured as the percentage of real-world objects modeled in the integrated schema that can be found in the sources. Therefore, the *schema completeness* criterion is the number of concepts provided by the schema with respect to the application domain.

Type Consistency

[1] **#** denotes the expression "*Number of*"

Received December 14, 2007.

Type consistency is the extent in which the attributes corresponding to the same real world concept are represented with the same data type across all schemas of a data integration system.

Minimality

Minimality is the extent in which the schema is compactly modeled and without redundancies. In our point of view, the minimality concept is very important to data integration systems because the integrated schema generated by the system may have redundancies. The key motivation for analyzing minimality is the statement that the more minimal the integrated schema is, the least redundancies it contains, and, consequently, the more efficient the query execution becomes [18]. Thus, we believe that our minimality analysis will help decreasing the extra time spent by mediators accessing to unnecessary information represented by redundant schema elements.

In this paper we discuss the use of minimality criterion analysis in a data integration system, when related to the system's schemas. This criterion defines that an schema element has good quality if it has no redundancies.

The quality analysis is performed by a software module called *IQ Manager* or *Information Quality Manager* which may be attached to a data integration system. At the moment of integrated schema generation or update, this module proceeds with the criteria assessment and then, according to the obtained IQ scores may execute adjustments over the schema to improve its design and, consequently, the query execution. This last step of schema tuning is executed after the IQ evaluation.

The paper is organized as follows: in section 2 we discuss Information Quality (IQ) and its use in data integration; section 3 discusses the formalism used for schemas representation; in section 4 we discuss minimality formal specification for the schema context; section 5 presents the schema improvement algorithm; section 6 presents the practical results obtained with the proposal implementation and section 7 discuss the final considerations about the mentioned topics.

2. Related Works

It has long been recognized that IQ is described or analyzed by multiple attributes or dimensions. During the past years, more and more dimensions and approaches were identified in several works [14, 26, 35, 2].

Naumann and Leser [26] define a framework addressing the IQ of query processing in a data integration system. This approach proposes the interleaving of query planning with quality considerations and creates a classification with twenty two dimensions divided into three classes: one related to the user preferences, the second class concerns the query processing aspects and the last one is related to the data sources.

The work proposed by Herden [14] deals with measuring the quality of conceptual database schemas. In this approach, given a quality criterion, the schema is reviewed by a specialist in the mentioned criterion.

In [35] the authors propose IQ evaluation for data warehouse schemas focusing on the analyzability and simplicity criteria

Some relevant works [5, 29, 24] are concerned with addressing IQ issues in data integration systems. Peralta in [29] proposes addressing the problem of data quality evaluation by a framework which is based on a graph model of the data integration system. The system is modeled as a workflow represented by a graph in which the activities perform the different tasks that extract, transform and convey data to users. It was presented an experiment with the data freshness IQ criteria. The work described in [24] also uses the activities graph representation for the integration system defined in [29]. The author defined the actual values of the quality properties at the sources, and at the integrated system there are the expected values of these properties. The focus of this work is the definition of strategies for the problem of managing the changes in the quality of sources, i.e. the management of the consequences that source quality changes may have on the system quality. Again, in this work, there is no reference about specific criteria for schema quality, only for the data-related criteria freshness and accuracy.

The proposal discussed in [5] is a data integration system with features to improve the user query processing. One of the optimization resources is the use of IQ criteria for selectively materialize data into a local repository.

Other relevant topic to consider in IQ and data integration is the set of quality criteria for schemas. These are critical due the importance of the integrated and data sources schemas for query processing. Some works are related to IQ aspects of schema equivalence and transformations, as in [2], where the authors exploit the use of normalization rules to improve IQ in conceptual database schemas.

We have found many works concerning IQ related to aspects of data integration. Some of them are concerning schema quality. But we have not find works related to investigate the impacts of minimality score in those schemas.

Our proposition is centered in IQ analysis for schemas of data integration systems. The differential of our approach is the proposal of processes of schema management associated with the minimality criterion examination features to obtain improvements in schema design and query execution.

3. Schema Representation

In data integration systems, the user submits queries to an *integrated schema*, which is a set of views, over a number of data sources, designed for a particular data integration application. Each source publishes a data source schema with the representation of its contents. The data integration system must reformulate a user query into queries that refers directly to schemas on the sources. To the reformulation step, the data integration system requires a set of correspondences, called *schema mappings.* Commonly, data integration systems use XML to represent the data and XML Schema to represent schemas . To provide a high-level abstraction for XML Schema elements [30], we use a conceptual data model, called X-Entity [20, 21] described in what follows. We also present the schema mappings with this notation.

3.1 X-Entity Model

The X-Entity model is an extension of the ER model [9], i.e., it extends it with additional features to represent XML schemas. The main concept of the model is the entity type, which represents the structure of XML elements composed by other elements and attributes. An X-Entity schema S is denoted by $S = (E,R)$, where E is a set of entity types and R is a set of relationship types.

- *Entity type*: an entity type E, denoted by $E(\{A_1,\ldots,A_n\},\{R_1,\ldots,R_m\})$, is made up of an entity name E, a set of attributes $\{A_1,\ldots,A_n\}$ and relationships $\{R_1,\ldots,R_m\}$. The attributes $\{A_1,\ldots,A_n\}$ represent either XML attributes or simple XML elements. In X-Entity diagrams, entity types are rectangles.

- *Containment relationship*: a containment relationship between two entity types E and E_1, specifies that each instance of E contains instances of E_1. It is denoted by $R(E,E_1,(min,max))$, where R is the relationship name and (min,max) are the minimum and the maximum number of instances of E_1 that can be associated with an instance of E.

- *Reference relationship*: a reference relationship, denoted by $R(E_1,E_2, \{A_{11},\ldots,A_{1n}\}, \{A_{21},\ldots,A_{2n}\})$, where R is the name of the relationship where the entity type E_1 references the entity type E_2. $\{A_{11},\ldots,A_{1m}\}$ and $\{A_{21},\ldots,A_{2n}\}$ are referencing attributes between E_1 and E_2 such that the value of A_{1i}, $1 \leq i \leq n$, in E_1 matches a value of A_{2i}, $1 \leq i \leq n$, in E_2.

3.2 Schema Mappings

There are several types of schema mappings to formally describe the associations between the concepts of X-Entity schemas [36, 23]. We consider an X-Entity *element* as an entity type, a relationship type or an attribute:

Entity schema mappings: if E_1 and E_2 are entity types, the schema mapping $E_1 \equiv E_2$ specifies that E_1 and E_2 are semantically equivalent, i.e., they describe the same real world concept and they have the same semantics.

Attribute schema mappings: are the mappings among attributes of semantically equivalent entities. The mapping $E_1.A_1 \equiv E_2.A_2$ indicates that A_1 and A_2 are semantically equivalent.

Path mappings: specify special types of mappings between attributes and subentities of semantically equivalent entity types with different structures.

Before defining a path mapping, it is necessary to define two concepts: *link* and *path*. A link between two X-Entity elements X_1 and X_2 $(X_1.X_2)$ occurs if X_2 is an attribute of the entity type X_1, or X_1 is an entity of the relationship type X_2 (or vice-versa).

If there is a multiple link, it is called a *path*. In this case it may occurs a normal path, $X_1.\ldots.X_n$ or an inverse path $(X_1.X_2.\ldots.X_n)^{-1}$. Any X-entity element is represented by paths. A path mapping can occur in four cases (assuming P_1 and P_2 as two paths):

Case 1: $P_1 = X_1.X_2...X_n$ and $P_2 = Y_1.Y_2...Y_m$, where $X_1 \equiv Y_1$. The mapping $P_1 \equiv P_2$ specifies that the entity types X_n and Y_m are semantically equivalent.

Case 2: $P_1 = X_1.X_2...X_n.A$ and $P_2 = Y_1.Y_2....Y_m.A'$, where $X_1 \equiv Y_1$. The mapping $P_1 \equiv P_2$ specifies that $A \in X_n$ and the attribute $A' \in Y_m$ are semantically equivalent.

Case 3: $P_1 = X_1.X_2...X_n$ and $P_2 = (Y_1.Y_2...Y_n)^{-1}$, where $X_1 \equiv Y_n$. The mapping $P_1 \equiv P_2$ specifies that the entity types X_n and Y_1 are semantically equivalent.

Case 4: $P_1 = X_1.X_2...X_n.A_k$ and $P_2 = (Y_1.Y_2...Y_n)^{-1}.A_k'$, where $X_1 \equiv Y_n$. The mapping $P_1 \equiv P_2$ specifies that the attribute $A_k \in X_n$ and the attribute $A_k' \in Y_1$ are semantically equivalent. Consider the integrated and data source schemas in Figure 1.

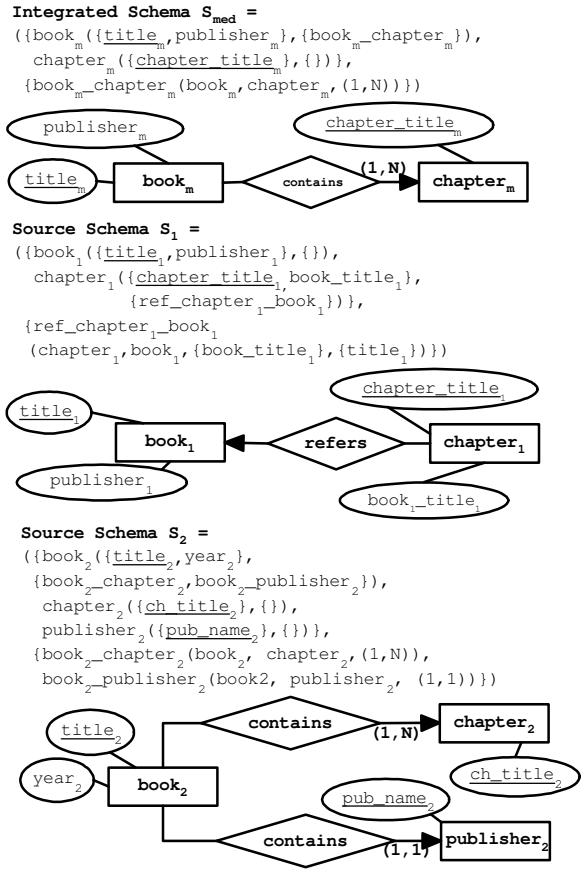

Figure 1. X-Entity Schemas

The Table 2 presents the relevant schema mappings identified to compute $book_m$ and $chapter_m$. The mappings specify the semantic equivalences between the integrated and data source schema elements.

Table 2. Schema mappings between the integrated schema S_{med} and schemas S_1 and S_2

MP1: $book_m \equiv book_1$
MP2: $book_m.title_m \equiv book_1.title_1$
MP3: $book_m.publisher_m \equiv book_1.publisher_1$
MP4: $chapter_m \equiv chapter_1$
MP5: $chapter_m.chapter\_title_m \equiv chapter_1.chapter\_title_1$
MP6: $book_m.book_m\_chapter_m.chapter_m \equiv (chapter_1.chapter\_ref\_book_1.book_1)^{-1}$
MP7: $book_m \equiv book_2$
MP8: $book_m.title_m \equiv book_2.title_2$
MP9: $chapter_m \equiv chapter_2$
MP10: $book_m.book_m\_chapter_m.chapter_m \equiv book_2.book_2\_chapter_2.chapter_2$
MP11: $chapter_m.chapter\_title_m \equiv chapter_2.ch\_title_2$
MP12: $book_m.publisher_m \equiv book_2.book_2\_publisher_2.publisher_2.pub\_name_2$

In data integration, the mappings are essential to assure the query processing over integrated schema. We assume that the mappings and schema elements equivalences are already defined automatically by the system or even manually by advanced users. It is very important to point that our work is not concerned with semantic similarities. The proposed IQ environment was designed to be included in an existent data integration system with all the schemas and mappings already created. Thus, we assume that the IQ module has access to a pre-existent set of semantic mappings between the data integration schemas.

Our proposition, centered in IQ analysis for schemas in data integration systems, has goals of query optimization and it is detailed in the following sections. The system uses the Seth's similarity scale for define schema equivalences [34]. Particularly, in the environment used to experiment our approach [5, 20, 21, 22], a schema matcher component is responsible to maintain equivalencies and mappings among sources and integrated schema.

4. The Minimality Criterion

The key motivation for analyzing minimality is the statement that the more minimal the integrated schema is, the least redundancies it contains, and, consequently, the more efficient the query execution becomes [18]. Thus, we have based our analysis in the measurement of minimality to help decreasing the extra time spent by mediator with access to unnecessary information represented by redundant schema elements.

It is important to notice that the proposed approach is not only to be applied in X-Entity schemas. The minimality IQ may be useful in any integrated schema to minimize problems resulting from schema integration processes, for example, to have semantically equivalent concepts represented more than once in one schema.

4.1 Definitions

It is necessary to consider an existent data integration system. More formally, a data integration system is defined as follows:

Definition 1 – Data Integration System (Ð):

A data integration system is a tuple, $\text{Ð} = \langle \delta, S_m \rangle$ where: δ is the set of S_i data sources schemas, i.e. $\delta = \langle S_1, S_2, \ldots, S_w \rangle$, where w is the number of data sources in Ð and S_m is the integrated schema, generated by modules of Ð. In Ð, the following statements are true:

- S_m is a X-Entity integrated schema such as $S_m = \langle E_1, E_2, \ldots, E_{n_m} \rangle$ where E_k is a mediation entity $(1 \leq k \leq n_m)$, and n_m is the number of entities in S_m;
- $\forall E_k \in S_m$, $E_k(\{A_{k1}, A_{k2}, \ldots, A_{ka_k}\}, \{R_{k1}, R_{k2}, \ldots, R_{kr_k}\})$, where $\{A_{k1}, A_{k2}, \ldots, A_{ka_k}\}$ is the set of attributes of E_k, $(a_k > 0)$; $\{R_{k1}, R_{k2}, \ldots, R_{kr_k}\}$ is the set of relationships of E_k, $(r_k \geq 0)$.
- If X_1 and X_2 are schema elements (attributes, relationships or entities), the schema mapping $X_1 \equiv X_2$ specifies that

X_1 and X_2 are *semantically equivalent*, i.e., they describe the same real world concept and have the same semantics.

Every information system (even a data integration one) is constructed from a number of requirements. Moreover, embedded in this set of requirements is the application domain information [19], very important to schemas construction.

In data integration context, we define a schema as *redundant* if it has occurrences of redundant entities and/or relationships. We introduce the definitions 2 to 5.

Definition 2 – Redundant attribute in a single entity:

An attribute A_{ki} of entity E_k, is *redundant*, i.e., $\mathbf{Red(A_{ki}, E_k)=1}$, if $\exists E_k.A_{kj}$, $j \neq i$, $A_{kj} \in \{A_{k1}, A_{k2}, \ldots, A_{ka_k}\}$ such as $E_k.A_{ki} \equiv E_k.A_{kj}$, $1 \leq i, j \leq a_k$

Definition 3 – Redundant attribute in different entities:

An attribute A_{ki} of the entity E_k, $A_{ki} \in \{A_{k1}, A_{k2}, \ldots, A_{ka_k}\}$ is redundant, i.e. $\mathbf{Red(A_{ki}, E_k) = 1}$, if: $\exists E_o$, $o \neq k$, $E_o \in S_m$, $E_k \equiv E_o$, $E_o(\{B_{o1}, B_{o2}, \ldots, B_{oa_o}\})$, B_{oj} are attributes of E_o and $\exists E_o.B_{oj}$, $B_{oj} \in \{B_{o1}, B_{o2}, \ldots, B_{oa_o}\}$ such as $E_k.A_{ki} \equiv E_o.B_{oj}$, $1 \leq i \leq a_k$, $1 \leq j \leq a_o$.

If for an attribute A_{ki} of entity E_k, $Red(A_{ki}, E_k)=0$, we say that A_{ki} is *non-redundant*.

Definition 4 – Entity Redundancy Degree:

An entity E_k has a positive redundancy degree in schema S_m, i.e. $Red(E_k, S_m)>0$, if E_k has at least one redundant attribute. The redundancy degree is calculated by the following formula:

$$Red(E_k, S_m) = \frac{\sum_{i=1}^{a_k} Red(A_{ki}, E_k)}{a_k}, \quad (1)$$

where

$\sum_{i=1}^{a_k} Red(A_{ki}, E_k)$ is the number of redundant attributes in E_k and;

a_k is the total number of attributes in E_k.

Definition 5 – Redundant Relationship:

Consider a relationship $R \in S_m$ between the entities E_k and E_y represented by the path $E_k.R.E_y$, $R \in \{R_{k1}, \ldots, R_{kr_k}\}$ and $R \in \{T_{y1}, \ldots, T_{yr_y}\}$, where $\{R_{k1}, \ldots, R_{kr_k}\}$ is the set of relationships of E_k and $\{T_{y1}, \ldots, T_{yr_y}\}$ is the set of relationships of E_y.

The relationship R connects E_k and E_y if and only if $R \in \{R_{k1}, \ldots, R_{kr_k}\}$ and $R \in \{T_{y1}, \ldots, T_{yr_y}\}$.

We define R as a *redundant relationship* in S_m, i.e. $Red(R, S_m) = 1$ if:

$\exists P_1$, $P_1 = E_k.R_j.\ldots.T_s.E_y$, P_1 is a path with

$R_j \in \{R_{k1}, \ldots, R_{kr_k}\}$ and $T_s \in \{T_{y1}, \ldots, T_{yr_y}\}$,

such as $P_1 \equiv R$.

In other words, a relationship between two entities is redundant if there are other semantically equivalent relationships which paths are connecting the same two entities.

It is important to say that a relationship equivalence is determined by a path equivalence, i.e., two relationships are semantically equivalents if their paths are also semantically equivalent.

In a redundancy analysis, where $\mathbf{E_1 \equiv E_2}$, we must decide if $\mathbf{Red(E_1, S_m) = 1}$ or $\mathbf{Red(E_2, S_m) = 1}$, because both situations are possible. However, only one element must be marked as redundant and removed, while the other has to be kept in the schema to assure that domain information will not be lost. In our approach we use some issues to decide which one of two redundant elements is marked and removed, as shown in section 5.

We agree with the work presented in [45], where Zhang states that redundancy is an *asymmetric* metric. He states that an element E_j may cause other element E_k to be viewed as redundant, but if the order is reversed, E_k may *not* cause E_j to be assigned as redundant, as it would be in a symmetric concept. A simple example is the case of an entity E_1, entirely contained in other entity E_2, E_1 may be viewed as redundant but E_2 may not.

4.2 Minimality

A schema is minimal if all of the domain concepts relevant for the application are described only once [18, 39, 31, 25].

Thus, we can say that the minimality of a schema is the degree of absence of redundant elements in the schema. Likewise our point of view, Kesh [18] argues that a more minimal (or *concise*) schema will make itself more efficient, and consequently improves the performance of operations and queries over it.

To measure the minimality, we must first determine the redundancy degree of the schema. To each one of the next redundancy definitions (6 and 7), we assume the following:

i) n_{rel} is the total number of relationships in S_m;

ii) n_m is the total number of entities in S_m;

iii) r_k is the number of relationships of each entity E_k in S_m;

Definition 6 – Entity Redundancy of a Schema:

The total entity redundancy of a schema S_m is computed by the formula:

$$ER(S_m) = \frac{\sum_{k=1}^{n_m} Red(E_k, S_m)}{n_m} \quad (2),$$

where $Red(E_k, S_m)$ is the redundancy degree of each E_k in S_m.

Definition 7 – Relationship Redundancy of a Schema:

The relationship redundancy degree of S_m is measured by the equation:

$$RR(S_m) = \frac{\#Red(R, S_m)}{n_{rel}} \quad (3),$$

where $\#Red(R, S_m)$ is the number of redundant relationships in S_m as stated in Definition 5.

Definition 8 – Schema Minimality:

We define the overall redundancy of a schema in a data integration system as the sum of the aforementioned redundancy values: entities (ER) and relationships (RR), by the formula:

$$Mi_{S_m} = 1 - [ER(S_m) + RR(S_m)] \quad (4)$$

4.3 Example

As an example of minimality evaluation, assume the redundant schema of Figure 2. The entity $artist_m$, is redundant because it is semantically equivalent to $actor_m$ and all its attributes have a semantically equivalent correspondent in $actor_m$.

The relationship $movie_m\_artist_m$ is also redundant because it has a semantically equivalent relationship $movie_m\_actor_m$ and $actor_m \equiv artist_m$. The schema minimality value will be obtained as in Figure 3.

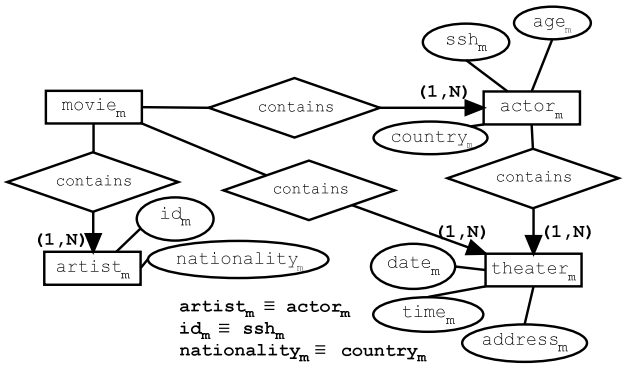

Figure 2. Schema with redundant elements

```
Red(movie_m,S_m) = 0
Red(actor_m,S_m) = 0
Red(theater_m,S_m) = 0
Red(artist_m,S_m)= 1
ER(S_m)= 1/(4 + 4) = 0,125
RR(S_m)= 1/(4 + 4) = 0,125
Mi(S_m)= 1 -(0,125 + 0.125) = 0,75
```

Figure 3. Schema minimality score

The minimality of schema S_m is 75%, what means that the schema has 25% of redundancy that can possibly be eliminated.

5. Schema IQ Improvement

After detecting the schema IQ anomalies, it is possible to restructure it to achieve better IQ scores [2]. In order to improve minimality scores, redundant elements must be removed from the schema. We proposed schema improvement actions specified in the algorithm of Table 3.

The condition of minimality = 1 is the ideal case where the schema is minimal, and this can occur when all schema redundancies are eliminated.

The detection of redundant elements processes are executed in steps 2, 4 and 6, already described in previous definitions. Redundancies elimination in steps 3, 5 and 7 are discussed in next sections.

Table 3. Schema improvement algorithm

1	Calculate minimality score and if minimality = 1, then stop;
2	Search for fully redundant entities in S_m;
3	If there are fully redundant entities then eliminate the redundant entities from S_m;
4	Search for redundant relationships in S_m;
5	If there are redundant relationships then eliminate the redundant relationships from S_m;
6	Search for redundant attributes in S_m;
7	If there are redundant attributes then eliminate the redundant attributes from S_m;
8	Go to Step 1

5.1 Redundant Entities Elimination

After removing a redundant entity E, its relationships must be relocated to a semantic equivalent remaining entity. When removing a redundant entity E_1 ($E_1 \equiv E_2$), the *IQ Manager* transfers the relationships of E_1 to the remaining equivalent entity E_2. Three different situations may occur when moving a relationship R_x, $R_x \in E_1$:

i) if $R_x \in E_2$ then R_x is deleted because it is no longer necessary;

ii) if $R_x \notin E_2$ but $\exists R_y$, $R_y \in E_2$ such as $R_x \equiv R_y$ then R_x is deleted;

iii) if $R_x \notin E_2$ and there is no R_y, $R_y \in E_2$ such as $R_x \equiv R_y$, then R_x is connected to E_2.

The first and second situations are not supposed to cause any other schema modification besides the entity deletion. The third case needs more attention, once redundant relationships of the removed entity have to be relocated as stated in the following.

Definition 9 – Substitute Entity:

E_k is a fully redundant entity, if and only if $Red(E_k, S_m)=1$ and E_k has at least one *Substitute Entity* E_s, i.e. $Subst(E_k) = E_s$, such as:

- $E_k(\{A_{k1}, \ldots, A_{ka_k}\}, \{R_{k1}, \ldots, R_{kr_k}\})$ A_{kx} are attributes and R_{ky} are relationships of E_k and;
- $E_s(\{A_{s1}, \ldots, A_{sa_s}\}, \{R_{s1}, \ldots, R_{sr_s}\})$ A_{sz} are attributes and R_{st} are relationships of E_s and
- $E_k \equiv E_s$ and $\forall E_k.A_{ki} \in \{A_{k1}, \ldots, A_{ka_k}\}$, $\exists E_s.A_{sj} \in \{A_{s1}, \ldots, A_{sa_s}\}$ with $E_k.A_{ki} \equiv E_s.A_{sj}$.

An entity E_k is considered fully redundant when all of its attributes are redundant, i.e. $\mathbf{Red(E_k, S_m)=1}$ and it has a substitute entity E_s in S_m. All the attributes of E_k are contained in E_s. E_k may be removed from the original schema S_m without lost of relevant information if it is replaced by its *substitute entity* E_s. Any existing relationship from E_k may be associated to E_s.

Definition 10 – Relationship Relocation:

In a schema S_m, if $Subst(E_k)=E_s$, then E_k can be eliminated from S_m. In this case, in order to do not lose any information, E_k relationships may be relocated in S_m. It is possible to relocate the relationships from E_k to E_s according to the following rules, i.e. $\forall E_k.R_{kj}$:

i. If $E_k.R_{kj} \in \{R_{s1}, \ldots, R_{sr_s}\}$ then R_{kj} must be deleted because it is no longer useful;

ii. If $E_k.R_{kj} \notin \{R_{s1}, \ldots, R_{sr_s}\}$ but $\exists E_s.R_{sp}$, such that $E_k.R_{kj} \equiv E_s.R_{sp}$ then $E_k.R_{kj}$ must be deleted because it has an equivalent relationship in E_s;

iii. If $E_k.R_{kj} \notin \{R_{s1}, \ldots, R_{sr_s}\}$ and $\nexists E_s.R_{sp}$ such as $E_k.R_{kj} \equiv E_s.R_{sp}$ then, E_s is redefined as $E_s(\{A_{s1}, \ldots, A_{sa_s}\}, \{R'_{s1}, \ldots, R'_{sr_s}\})$, A_{sz} are attributes and R'_{st} are relationships of E_s and $\{R'_{s1}, \ldots, R'_{sr_s}\} = \{R_{s1}, \ldots, R_{sr_s}\} \cup R_{kj}$.

The relationship relocation is illustrated in Figure 3. In the Figure 3, the redundant elements are represented in grey.

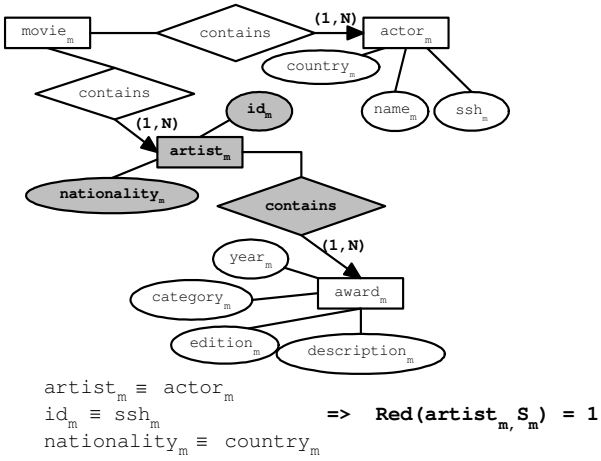

Figure 3. Redundant entity elimination

The fully redundant entity $artist_m$ (with its attributes) is removed and it is substituted by the semantically equivalent $actor_m$. Consequently, the relationship $movie_m\_artist_m$ may be deleted and it is replaced by the remaining equivalent relationship $movie_m\_actor_m$.

The relationship $artist_m\_award_m$ is relocated to $actor_m$, turning into the new relationship $actor_m\_award_m$. With these operations, it is possible to obtain the non redundant schema represented in Figure 4.

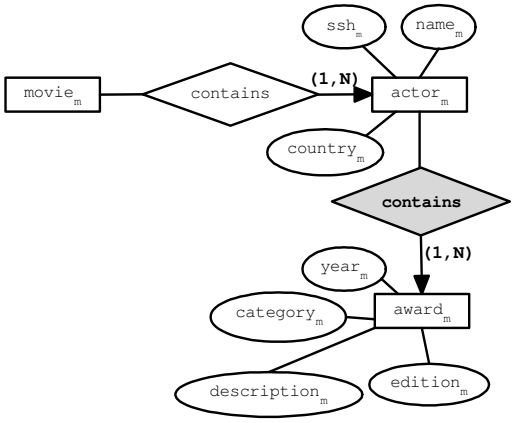

Figure 4. Schema after redundant entity elimination

5.2 Redundant Relationships Elimination

After removing redundant entities and performing the necessary relationship relocations, the *IQ Manager* is supposed to analyze if there are remaining redundant relationships to eliminate them. This can be accomplished by purely deleting from the schema, the relationships identified as redundant.

After eliminating the redundant relationships the schema becomes with no relationship redundancies and do not have had lost of relevant information.

5.3 Redundant Relationships Elimination

The last step of schema improvement algorithm consists in investigating and eliminating remaining redundant attributes in schema. Similarly to the redundant relationships removal step, these attributes may merely be deleted from schema. This occurs because the schema always has semantically equivalent attributes to substitute the redundant ones. After executing the schema improvement steps, the *IQ Manager* can recalculate and analyze minimality scores in order to determine if the desired IQ is accomplished.

6. Experimentation Results

We implemented the IQ Manager in an existing mediator-based data integration system. It is a software module that executes minimality analysis and schema improvement actions.

At the moment of integrated schema generation or update, the *IQ Manager* proceeds with the criterion assessment and then, according to the obtained scores, executes adjustments over the schema presented in Section 5. More details about the data integration system can be found in [5].

The module was written in Java and the experiment used two databases – MySQL and PostgreSQL – to store the data sources. As mentioned before, the data in the system are represented with XML and the schemas with XML Schema. In our experimentation the following steps were executed:

(i) the queries were submitted over an integrated schema with a some redundancy and the execution times were measured;

(ii) the redundancy elimination algorithm was executed over the redundant integrated schema generating a completely minimal schema;

(iii) the same queries used in step (i) were executed.

The results obtained with these experiments have been satisfactory since query performance was improved. We used a real world data integration application in health care domain, the mediator works with two data sources: one with data of a public hospital and the other with data obtained from video-conferencing sessions of real-time consultations between medical specialists in different locations.

There are portions of the data source schemas in Figures 5 and 6. The redundant integrated schema is presented in Figure 7.The schema mappings between the data sources and the integrated schema are listed in Table 4.

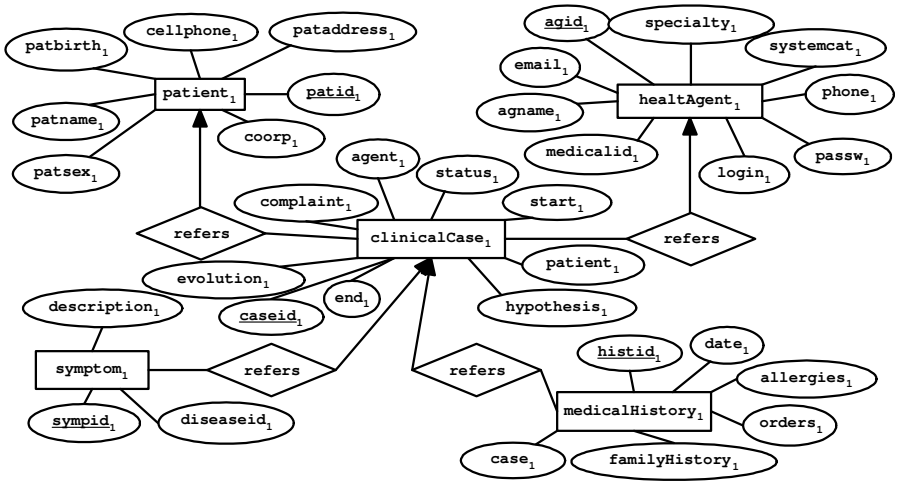

Figure 5. Schema of public hospital data source (S_1)

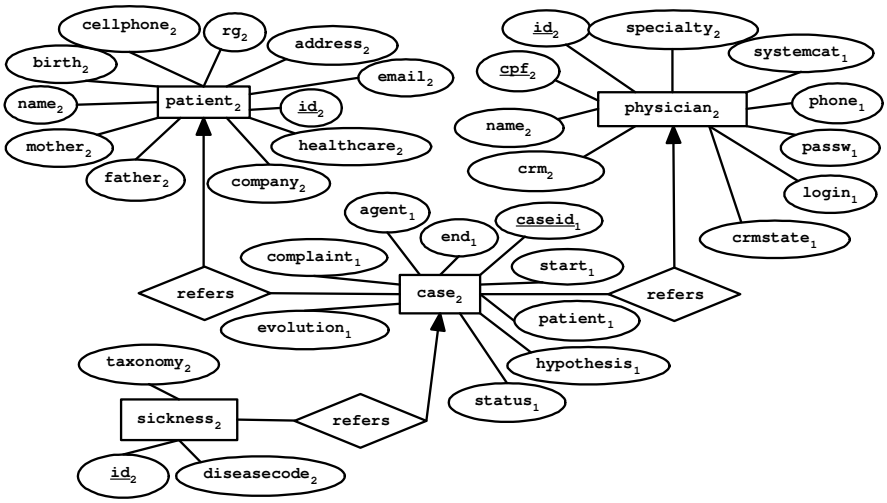

Figure 6. Schema of telemedicine data source (S_2)

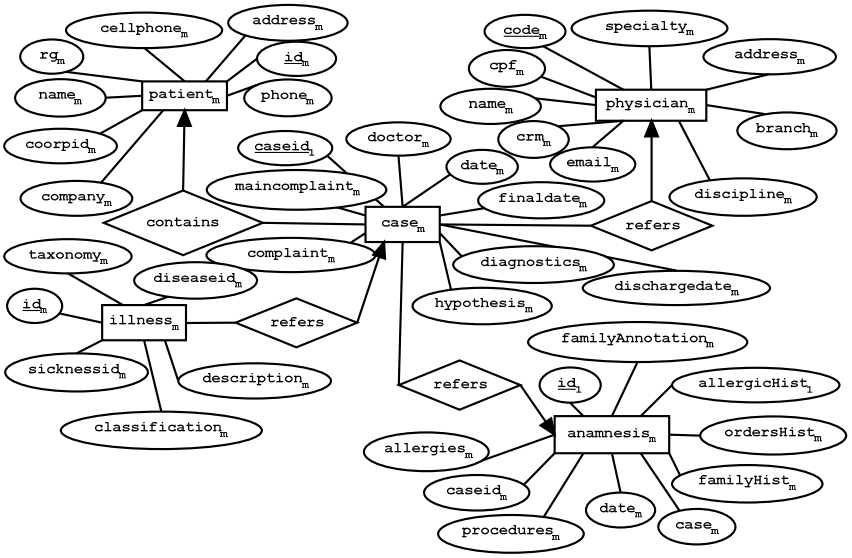

Figure 7. Redundant integrated schema (S_m)

Table 4. Schema mappings between the redundant integrated schema S_m and the source schemas S_1 and S_2

MP_1: $case_m \equiv clinicalCase_1$
MP_2: $case_m \equiv case_2$
MP_3: $physician_m \equiv healthAgent_1$
MP_4: $physician_m \equiv physician_2$
MP_5: $patient_m \equiv patient_1$
MP_6: $patient_m \equiv patient_2$
MP_7: $ilness_m \equiv disease_2$
MP_8: $ilness_m \equiv sickness_2$
MP_9: $anamnesis_m \equiv medicalHistory_1$
MP_{10}: $case_m.date_m \equiv clinicalCase_1.start_1$
MP_{11}: $case_m.diagnostics_m \equiv clinicalCase1.hypothesis_1$
MP_{12}: $case_m.doctor_m \equiv clinicalCase_1.agent_1$
MP_{13}: $case_m.finaldate_m \equiv clinicalCase_1.end_1$
MP_{14}: $case_m.dischargedate_m \equiv clinicalCase_1.end_1$
MP_{15}: $case_m.complaint_m \equiv clinicalCase_1.maincomplaint_1$
MP_{16}: $case_m.hypothesis_m \equiv clinicalCase_1.hypothesis_1$
MP_{17}: $case_m.date_m \equiv case_2.startdate_1$
MP_{18}: $case_m.diagnostics_m \equiv case_2.diagnostics_2$
MP_{19}: $case_m.doctor_m \equiv case_2.doctor_2$
MP_{20}: $case_m.hypothesis_m \equiv case_2.diagnostics_2$
MP_{21}: $case_m.maincomplaint_m \equiv case_2.complaint_2$
MP_{22}: $case_m.complaint_m \equiv case_2.complaint_2$
MP_{23}: $physician_m.name_m \equiv healthAgent_1.agname_1$
MP_{24}: $physician_m.specialty_m \equiv healthAgent_1.specialty_1$
MP_{25}: $physician_m.email_m \equiv healthAgent_1.email_1$
MP_{26}: $physician_m.code_m \equiv healthAgent_1.agid_1$
MP_{27}: $physician_m.branch_m \equiv healthAgent_1.specialty_1$
MP_{28}: $physician_m.crm_m \equiv healthAgent_1.medicalid_1$
MP_{29}: $physician_m.name_m \equiv physician_2.name_2$
MP_{30}: $physician_m.address_m \equiv physician_2.address_2$
MP_{31}: $physician_m.specialty_m \equiv physician_2.specialty_2$
MP_{32}: $physician_m.code_m \equiv physician_2.cpf_2$
MP_{33}: $physician_m.cpfm \equiv physician_2.cpf_2$
MP_{34}: $physician_m.branch_m \equiv physician_2.specialty_2$
MP_{35}: $physician_m.discipline_m \equiv physician_2.specialty_2$
MP_{36}: $physician_m.email_m \equiv physician_2.email_2$
MP_{37}: $physician_m.crm_m \equiv physician_2.crm_2$
MP_{38}: $patient_m.name_m \equiv patient_1.patname_1$
MP_{39}: $patient_m.address_m \equiv patient_1.pataddress_1$
MP_{40}: $patient_m.company_m \equiv patient_1.coorp_1$
MP_{41}: $patient_m.phone_m \equiv patient_1.cellphone_1$
MP_{42}: $patient_m.coorpid_m \equiv patient_1.coorp_1$
MP_{43}: $patient_m.name_m \equiv patient_2.name_2$
MP_{44}: $patient_m.address_m \equiv patient_2.address_2$
MP_{45}: $patient_m.company_m \equiv patient_2.company_2$
MP_{46}: $patient_m.cellphone_m \equiv patient_2.cellphone_2$
MP_{47}: $patient_m.coorpid_m \equiv patient_2.company_2$
MP_{48}: $patient_m.rg_m \equiv patient_2.rg_2$
MP_{49}: $anamnesis_m.id_m \equiv medicalHistory_1.histid_1$
MP_{50}: $anamnesis_m.allergies_m \equiv medicalHistory_1.allergies_1$
MP_{51}: $anamnesis_m.caseid_m \equiv medicalHistory_1.case_1$
MP_{52}: $anamnesis_m.procedures_m \equiv medicalHistory_1.orders_1$
MP_{53}: $anamnesis_m.date_m \equiv medicalHistory_1.date_1$
MP_{54}: $anamnesis_m.case_m \equiv medicalHistory_1.case_1$
MP_{55}: $anamnesis_m.familyHist_m \equiv$

$medicalHistory_1.familyHistory_1$
MP_{56}: $anamnesis_m.ordersHist_m \equiv medicalHistory_1.orders_1$
MP_{57}: $anamnesis_m.allergicHist_m \equiv medicalHistory_1.allergies_1$
MP_{58}: $anamnesis_m.familyAnnotation_m \equiv medicalHistory_1.familyHistory_1$
MP_{59}: $case_m.case_m\_patient_m.patient_m \equiv clinicalCase_1.clinicalCase_1\_ref\_patient_1.patient_1$
MP_{60}: $case_m.case_m\_patient_m.patient_m \equiv case_2.case_2\_ref\_patient_2.patient_2$
MP_{61}: $case_m.case_m\_ref\_physician_m.physician_m \equiv clinicalCase_1.clinicalCase_1\_ref\_healthAgent_1.healthAgent_1$
MP_{62}: $case_m.case_m\_ref\_physician_m.physician_m \equiv case_2.case_2\_ref\_physician_2.physician_2$
MP_{63}: $ilness_m.id_m \equiv disease_1.id_1$
MP_{64}: $ilness_m.diseaseid_m \equiv disease_1.diseaseid_1$
MP_{65}: $ilness_m.description_m \equiv disease_1.description_1$
MP_{66}: $ilnessm.sicknessidm \equiv diseasel.diseaseid1$
MP_{67}: $ilness_m.id_m \equiv sickness_2.id_2$
MP_{68}: $ilness_m.diseaseid_m \equiv sickness_2.diseasecode_2$
MP_{69}: $ilness_m.taxonomy_m \equiv sickness_2.taxonomy_2$
MP_{70}: $ilness_m.sicknessid_m \equiv sickness_2.diseasecode_2$
MP_{71}: $ilness_m.taxonomy_m \equiv sickness_2.classification_2$
MP_{72}: $case_m.case_m\_ref\_ilness_m.ilness_m \equiv clinicalCase_1.clinicalCase_1\_ref\_symptom_1.symptom_1$
MP_{73}: $anamnesis_m.anamnesis_m\_ref\_case_m.case_m \equiv (clinicalCase_1.clinicalCase_1\_ref\_medicalHistory_1.medicalHistory_1)^{-1}$
MP_{74}: $case_m.case_m\_ref\_ilness_m.ilness_m \equiv case_2.case_2\_ref\_sickness_2.sickness_2$

After analyzing the schemas of Figures 5, 6 and 7 and schema mappings of Table 4, the IQ module calculates the following minimality values for the integrated schema: $Red(case_m, S_m) = 0.3750$; $Red(patient_m, S_m) = 0.1250$; $Red(physician_m, S_m) = 0.2778$; $Red(ilness_m, S_m) = 0.3333$.

These entity minimality scores result in a integrated schema with overall minimality degree of 72.22%. The 27.78% of redundancy can be completely eliminated by the algorithm presented in Section 5. The output of the IQ manager adjustment process is the minimal integrated schema presented in Figure 8 and non-redundant set of schema mappings in Table 5.

Table 5. Schema mappings between the minimal integrated schema S_m and the source schemas S_1 and S_2

MP_1: $case_m \equiv clinicalCase_1$
MP_2: $case_m \equiv case_2$
MP_3: $physician_m \equiv healthAgent_1$
MP_4: $physician_m \equiv physician_2$
MP_5: $patient_m \equiv patient_1$
MP_6: $patient_m \equiv patient_2$
MP_7: $ilness_m \equiv disease_2$
MP_8: $ilness_m \equiv sickness_2$
MP_9: $anamnesis_m \equiv medicalHistory_1$
MP_{10}: $case_m.date_m \equiv clinicalCase_1.start_1$
MP_{11}: $case_m.diagnostics_m \equiv clinicalCase1.hypothesis_1$
MP_{12}: $case_m.doctor_m \equiv clinicalCase_1.agent_1$
MP_{14}: $case_m.dischargedate_m \equiv clinicalCase_1.end_1$
MP_{15}: $case_m.complaint_m \equiv clinicalCase_1.maincomplaint_1$
MP_{17}: $case_m.date_m \equiv case_2.startdate_1$
MP_{18}: $case_m.diagnostics_m \equiv case_2.diagnostics_2$
MP_{19}: $case_m.doctor_m \equiv case_2.doctor_2$
MP_{22}: $case_m.complaint_m \equiv case_2.complaint_2$
MP_{23}: $physician_m.name_m \equiv healthAgent_1.agname_1$
MP_{24}: $physician_m.specialty_m \equiv healthAgent_1.specialty_1$
MP_{25}: $physician_m.email_m \equiv healthAgent_1.email_1$
MP_{26}: $physician_m.code_m \equiv healthAgent_1.agid_1$
MP_{28}: $physician_m.crm_m \equiv healthAgent_1.medicalid_1$
MP_{29}: $physician_m.name_m \equiv physician_2.name_2$
MP_{30}: $physician_m.address_m \equiv physician_2.address_2$
MP_{31}: $physician_m.specialty_m \equiv physician_2.specialty_2$
MP_{32}: $physician_m.code_m \equiv physician_2.cpf_2$
MP_{36}: $physician_m.email_m \equiv physician_2.email_2$
MP_{37}: $physician_m.crm_m \equiv physician_2.crm_2$
MP_{38}: $patient_m.name_m \equiv patient_1.patname_1$
MP_{39}: $patient_m.address_m \equiv patient_1.pataddress_1$
MP_{40}: $patient_m.company_m \equiv patient_1.coorp_1$
MP_{41}: $patient_m.phone_m \equiv patient_1.cellphone_1$
MP_{43}: $patient_m.name_m \equiv patient_2.name_2$
MP_{44}: $patient_m.address_m \equiv patient_2.address_2$
MP_{45}: $patient_m.company_m \equiv patient_2.company_2$
MP_{46}: $patient_m.cellphone_m \equiv patient_2.cellphone_2$
MP_{48}: $patient_m.rg_m \equiv patient_2.rg_2$
MP_{49}: $anamnesis_m.id_m \equiv medicalHistory_1.histid_1$
MP_{50}: $anamnesis_m.allergies_m \equiv medicalHistory_1.allergies_1$
MP_{51}: $anamnesis_m.caseid_m \equiv medicalHistory_1.case_1$
MP_{52}: $anamnesis_m.procedures_m \equiv medicalHistory_1.orders_1$
MP_{53}: $anamnesis_m.date_m \equiv medicalHistory_1.date_1$
MP_{55}: $anamnesis_m.familyHist_m \equiv medicalHistory_1.familyHistory_1$
MP_{59}: $case_m.case_m\_patient_m.patient_m \equiv clinicalCase_1.clinicalCase_1\_ref\_patient_1.patient_1$
MP_{60}: $case_m.case_m\_patient_m.patient_m \equiv case_2.case_2\_ref\_patient_2.patient_2$
MP_{61}: $case_m.case_m\_ref\_physician_m.physician_m \equiv clinicalCase_1.clinicalCase_1\_ref\_healthAgent_1.healthAgent_1$
MP_{62}: $case_m.case_m\_ref\_physician_m.physician_m \equiv case_2.case_2\_ref\_physician_2.physician_2$
MP_{63}: $ilness_m.id_m \equiv disease_1.id_1$
MP_{64}: $ilness_m.diseaseid_m \equiv disease_1.diseaseid_1$
MP_{65}: $ilness_m.description_m \equiv disease_1.description_1$
MP_{66}: $ilness_m.sicknessid_m \equiv disease_1.diseaseid_1$
MP_{67}: $ilness_m.id_m \equiv sickness_2.id_2$
MP_{68}: $ilness_m.diseaseid_m \equiv sickness_2.diseasecode_2$
MP_{69}: $ilness_m.taxonomy_m \equiv sickness_2.taxonomy_2$
MP_{70}: $ilness_m.sicknessid_m \equiv sickness_2.diseasecode_2$
MP_{72}: $case_m.case_m\_ref\_ilness_m.ilness_m \equiv clinicalCase_1.clinicalCase_1\_ref\_symptom_1.symptom_1$
MP_{73}: $anamnesis_m.anamnesis_m\_ref\_case_m.case_m \equiv (clinicalCase_1.clinicalCase_1\_ref\_medicalHistory_1.medicalHistory_1)^{-1}$
MP_{74}: $case_m.case_m\_ref\_ilness_m.ilness_m \equiv case_2.case_2\_ref\_sickness_2.sickness_2$

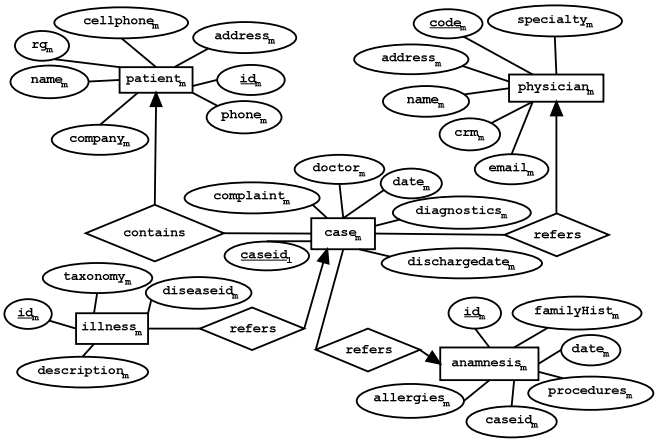

Figure 8. Minimal integrated schema (S_m)

To experiment our arguments and points of view, we choose four types of user queries and executed the same query over the redundant schema of Figure 7 and over the minimal schema of Figure 8. Each query was submitted five times, and its processing times were computed. In all of the four queries, the average execution time was lower when the integrated schema is minimal, as it can be seen in the user queries UQ1 to UQ4 presented as follows.

UQ1: simple selection

This user query is a simple selection asking for all the attributes of the mediation entity $anamnesis_m$. The graph of Figure 9, illustrates the execution times.

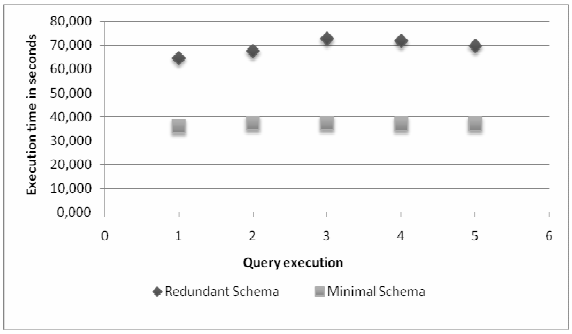

Figure 9. Execution times of a simple selection user query (UQ1)

The average execution time for the first query UQ1 was **69,372** seconds in a redundant schema versus **37,009** seconds for the same query submitted over a minimal schema. The query returned over 1.000 records. This represented a percentage gain of **46,65%**.

UQ2: selection with condition

This query is a selection asking for all the attributes of the $\texttt{anamnesis}_m$ entity with surgery procedures.

The graph of Figure 10, illustrates the set of execution times of UQ2 submitted over the redundant and the minimal schemas.

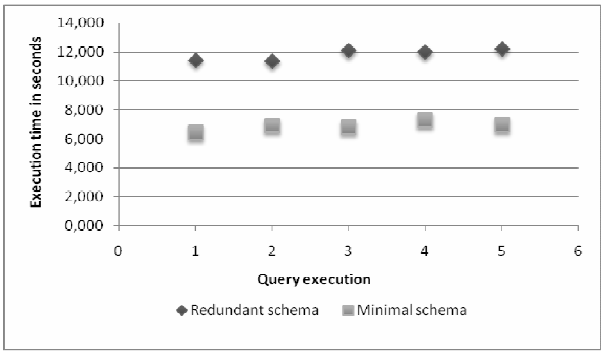

Figure 10. Execution times of a user query with condition (UQ2)

The average execution time was **11,831** seconds over the redundant schema and **6,897** seconds for the same query submitted over a minimal schema.

The query returned over 500 records. This represents a percentage gain of **41,70%.**

UQ3: **Join with two condition tests**

This query is a join between $\texttt{case}_m$ and $\texttt{patient}_m$ entities. It asks for case elements which patients live in city of "Recife" and state of "Pernambuco". The graph of Figure 11, illustrates the UQ3 execution times.

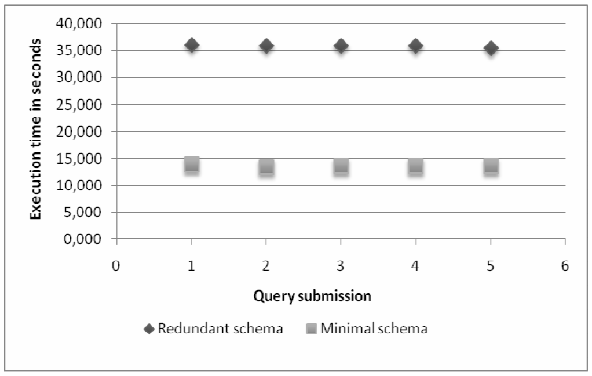

Figure 11. Execution times of a user query with containment relationship and condition (UQ3)

The average execution time in this case was **35,913** seconds in the first submission over the redundant schema versus **13,590** seconds for the same query submitted over the minimal schema. This represents a percentage improvement of **62,16%.**

UQ4: join with one condition

This query is a join between the entities case and physician, asking for the attributes of the cases which physicians have last name "Kaufmann". The graph of Figure 12, illustrates the execution times.

The average execution time in this case was **12,078** seconds in a redundant schema versus **8,325** seconds for the query submitted over a minimal schema. This represents a percentage gain of **31,07**%. Thus, this confirms the existence of improvements in query execution time.

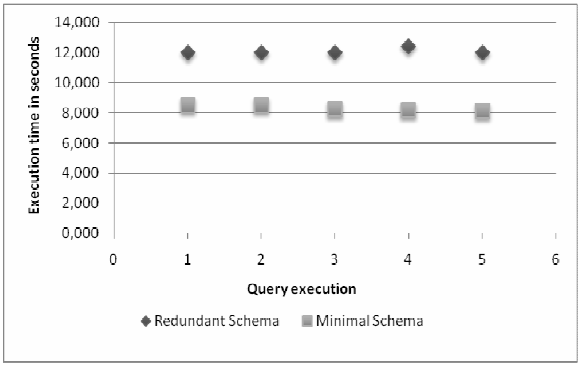

Figure 12. Execution times of a join (UQ4)

It is important to observe that in all of the tested cases, the results confirm the existence of improvements in query execution time. The execution times for queries over the minimal schema were significantly minor. The four query execution times are summarized in Table 6 and in Figure 13.

Table 6 Summary of query execution times

Query		Average Times (Sec)		Performance
		Redundant Schema	Minimal Schema	Gain(%)
UQ1	Simple selection	69,3720	37,0094	46,65%
UQ2	Selection with condition	11,8312	6,8970	41,70%
UQ3	Join with two	35,9128	13,5904	62,16%

	conditions			
UQ4	Join with one condition	12,0776	8,3248	31,07%
		Average Gain		**45,40%**

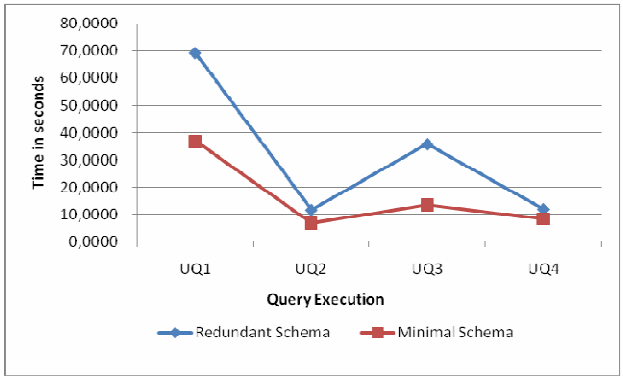

Figure 13. Summary of execution times of UQ

By comparing the final results, it is possible to see that the query performance was improved in an average time of **45,40%.**

7. Conclusions and Future Works

Data integration systems may suffer with lack of quality in their produced query results. They can be outdated, erroneous, incomplete, redundant and so on. To minimize the impact of these problems, we propose a quality approach that serves to analyze and improve the integrated schema definition and consequently, the query execution. The main contribution of the presented approach is the specification of minimality IQ criteria assessment methods for the maintenance of high quality integrated schemas with objectives of achieving better integrated query execution. We have implemented the IQ Manager in an existing data integration system module, to analyze the integrated schema minimality and to eliminate the redundant items. The detailed contributions are:

i. Consolidation of IQ usage in data integration systems through the classification of a set of criteria specifically selected for this kind of environment;

ii. Specification of a relevant schema IQ criterion, i.e. minimality, in the context of a data integration system;

iii. Analysis of system's elements according to the specified minimality criteria. We presented the IQ analysis of schemas associated with an algorithm for minimality improvement.

iv. The proposed approach was experimentally validated through specification, implementation and tests of the IQ Manager in a data integration system with a real health care application.

It is important to mention that we have found several works concerning IQ related to aspects of data integration and schemas. However, none of them were concerned to verify the impacts of minimality in those schemas.

As future work, we will formally describe and implement the algorithms to evaluate others schema related IQ criteria. We already have started the specification of the type consistency and completeness criteria using similar concepts as used in [10, 13, 40]. More detail about our work with other IQ criteria is presented in [6, 7, 8].

Possibly, we also may extend our IQ studies to analyze the quality impacts in information retrieval area.

References

[1] ANGELES P. and MACKINNON, L. Quality Measurement and Assessment Models including Data Provenance to grade Data sources. In proceedings of International Conference on Computer Science and Information Systems, Athens, Greece, 2005.

[2] ASSENOVA, P. and JOHANNESON, P. Improving Quality in Conceptual Modeling by the Use of Schema Transformations. In Proceedings 15th Int. Conf. of Conceptual Modeling (ER´96), Cottbus, Germany, 1996.

[3] BALLOU, D.P. and PAZER, H.L. Modeling Data and Process Quality in Multi-input, Multi-output Information Systems. *Management Science* 1985.

[4] BALLOU, D. P. and PAZER H.: Modeling Completeness versus Consistency Tradeoffs in Information Decision Contexts. IEEE Transactions on Knowledge and Data Engineering, 15(1), 2003.

[5] BATISTA, M. C., LÓSCIO, B. F. and SALGADO, A. C. Optimizing Access in a Data Integration System with Caching and Materialized Data. In Proceedings of 5th *ICEIS*, 2003.

[6] BATISTA, M. C. and SALGADO, A.C. Minimality Quality Criterion Evaluation for Integrated Schemas. In Proceedings of the Second International Conference on Digital Information Management (ICDIM'07). Lyon, France. 2007.

[7] BATISTA, M. C. and SALGADO, A.C. Data Integration Schema Analysis: An Approach with Information Quality. In Proceedings of the 12th International Conference on Information Quality (ICIQ 07). MIT, Massachusetts, USA. 2007.

[8] BATISTA, M. C. and SALGADO, A.C. Information Quality Measurement in Data Integration Schemas. In Proceedings of the 5th International Workshop on Quality in Databases at VLDB. Vienna, Austria. 2007.

[9] CHEN, P.P. The Entity-Relationship Model: Toward a Unified View of Data. ACM Transactions on Database Systems, 1976.

[10] DAI, B., T., KOUDAS, N., OOI, B. C., SRIVASTAVA, D. and VENKATASUBRAMANIAN, S. Column Heterogeneity as a Measure of Data Quality, in Proceedings of 1st Int'l VLDB Workshop on Clean Databases, 2006.

[11] GE, M. and HELFERT, M. A Review of Information Quality Research - Develop a Research Agenda. In Proceedings of the 12th International Conference on Information Quality (ICIQ 07). MIT, Massachusetts, USA. 2007.

[12] GERTZ, M.; TAMER OZSU, M.; SAAKE, G. and SATTLER, K.: "Report on the Dagstuhl Seminar: Data Quality on the Web". SIGMOD Record, 33(1), March 2004.

[13] HALEVY, A. Why Your Data Don't Mix. *ACM Queue, 3(8),* 2005.

[14] HERDEN, O. Measuring Quality of Database Schema by Reviewing - Concept, Criteria and Tool. In Proceedings of 5th Intl Workshop on Quantitative Approaches in Object-Oriented Software Engineering, 2001.

[15] JARKE, M. and Y. VASSILIOU. Data Warehouse Quality: A Review of the DWQ Project. in Proceedings of the 1997 Conference on Information Quality. Cambridge, MA, 299-313, 1997.

[16] JARKE, M., JEUSFELD, M. A., QUIX, C. and VASSILIADIS, P.: Architecture and Quality in Data Warehouses: An Extended Repository Approach. Inf. Syst. 24(3): 229-253, 1999.

[17] JIANG L., BORGIDA, A., TOPAGLOU T. and MYLOPOULOS J. Data Quality By Design: A Goal-Oriented Approach. In Proceedings of the 12th International Conference on Information Quality (ICIQ 07). MIT, Massachusetts, USA. 2007.

[18] KESH, S. Evaluating the Quality of Entity Relationship Models. Inform. Software Technology. 1995.

[19] KOTONYA, G and SOMMERVILLE, I. Requirements Engineering: Processes and Techniques. 1st Edition, Wiley & Sons, 1997.

[20] LÓSCIO, B. F., Managing the Evolution of XML-Based Mediation Queries. Tese de Doutorado. Curso de Ciência da Computação. Centro de Informática, UFPE, Recife, 2003.

[21] LÓSCIO, B. F., SALGADO, A. C. and GALVÃO, L. R.: Conceptual Modeling of XML Schemas, In Proceedings of Fifth International Workshop on Web Information and Data Management (WIDM 2003), New Orleans Louisiana USA, 102-105, 2003.

[22] LÓSCIO, B. F. and SALGADO, A. C.: Generating Mediation Queries for XML-based Data Integration Systems, In Proceedings of XVIII Brazilian Symposium on Databases (SBBD 2003), Manaus Amazonas Brazil, 2003.

[23] LOSCIO, B. F., COSTA, T. A., SALGADO, A. C. and FREITAS, J. S. Query Reformulation for an XML-based Data Integration System, In: The 21st Annual ACM Symposium on Applied Computing, ACM Press, Dijon, France, 2006.

[24] MAROTTA, A. and RUGGIA, R. Managing Source Quality Changes in Data Integration Systems. In proceedings of 2nd International Workshop on Data and Information Quality (DIQ'05), 2005.

[25] MOODY, D. Measuring the Quality of Data Models: An Empirical Evaluation of the Use of Quality Metrics in Practice, New Paradigms in Organizations, Markets & Society. In Proceedings of 11th European Conference on Information Systems, 2003.

[26] NAUMANN, F. and LESER, U. Quality-driven Integration of Heterogeneous Information Systems. In Proceedings of the 25th *VLDB*. 1999.

[27] NAUMANN, F. and ROLKER, C. Assessment Methods for Information Quality Criteria. In Proceedings of the Conference on International Quality (IQ00) Boston, 2000.

[28] NAUMANN F. From Databases to Information Systems – Information Quality Makes the Difference. In Proceedings of The International Conference on Information Quality, 2001.

[29] PERALTA, V., RUGGIA, R., KEDAD, Z. and BOUZEGHOUB, M. A Framework for Data Quality Evaluation in a Data Integration System. In Proceedings of 19º Simposio Brasileiro de Banco de Dados (SBBD'2004), 2004.

[30] PETERSON, D., BIRON, P. V., MALHOTRA, A. and SPERBERG-MCQUEEN., C. M. *XML Schema 1.1 Part 2: Data Types* – W3C Working Draft, 2006. http://www.w3.org/TR/xmlschema11-2/.

[31] PIATTINI, M., GENERO, M. and CALERO, C. Data Model Metrics. In Handbook of Software Engineering and Knowledge Engineering: Emerging Technologies, World Scientific, 2002.

[32] REDMAN T.C.: Data Quality for the Information Age. Artech House, 1996.

[33] SCANNAPIECO, M. Data Quality at a Glance. Datenbank-Spektrum *14*, 6–14. 2005.

[34] SHETH, A. and KASHYAP, V. So Far (Schematically) yet So Near (Semantically), In Proceedings of IFIP WG 2.6 Conference on Semantics of Interoperable Database Systems (Data Semantics 5), North Holland, Amsterdam, 1993.

[35] SI-SAID, S. C. and PRAT, N. Multidimensional Schemas Quality: Assessing and Balancing Analyzability and Simplicity, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2814, 140—151. 2003.

[36] SPACCAPIETRA, S. and PARENT, C. View Integration: a Step Forward in Solving Structural Conflicts. IEEE Transactions on Knowledge and Data Engineering, 6(2), 1994.

[37] STVILIA, B., GASSER, L., TWIDALE, M. B. and SMITH, L.C. A framework for information quality assessment. In Journal of the American Society for Information Science and Technology, 58(12), 2007.

[38] TAYI, G. K. and BALLOU, D. P. Examining Data Quality. Communications of the ACM 41(2), 1998.

[39] VARAS, M. Diseño Conceptual de Bases de Datos: Un enfoque Basado en la Medición de la Calidad", Actas Primer Workshop Chileno de Ingeniería de Software, Punta Arenas, 2001.

[40] WAND, Y. and WANG, R.Y. Anchoring Data Quality Dimensions in Ontological Foundations. Communications of the ACM 39(11), 86—95, 1996.

[41] WANG, R., KON, H. and MADNICK, S. Data Quality Requirements Analysis and Modeling, In Proceedings of 9th International Conference of Data Engineering, Vienna, Austria, 1993.

[42] WANG R.Y. and STRONG D.M. Beyond Accuracy: What Data Quality Means to Data Consumers. Journal of Management Information Systems, 12(4), 1996.

[43] WANG, R.Y. A Product Perspective on Total Data Quality Management. Communications of the ACM 41(2), 58--65. 1998.

[44] WIEDERHOLD, G., Mediators in the Architecture of Future Information Systems. *IEEE Computer*. 2, 1992.

[45] ZHANG, Y., CALLAN, J. and MINKA. T. Novelty and Redundancy Detection in Adaptive Filtering. In Proceedings of the 25 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 2002.

Author Biographies

Maria da Conceição Moraes Batista received her Bachelor degree and MS degree in computer science from Center for Informatics of Universidade Federal de Pernambuco – UFPE – (Brazil) in 1988 and 2003, respectively. Currently she is a PhD student in Center for Informatics of UFPE. She has over 10 years of software development. Her major research areas are information quality, databases, Web systems and software development.

Ana Carolina Salgado is currently an Associate Professor at the Center for Informatics of Universidade Federal de Pernambuco – UFPE – (Brazil). She obtained her Doctorate in Computer Science from the University of Nice (France) in 1988, her Master (1983) and her Bachelor degree (1980) in Computer Science from UFPE. Her main research interests are in the areas of databases, information integration on the Web and cooperative systems. Dr. Salgado has published over a hundred technical articles in conference proceedings and journals. In her academic activities she has advised 34 MSc and 6 PhD theses. She is member of the ACM and of the Brazilian Computer Society. She also was head of Center for Informatics and coordinator of the Computer Science undergraduate course at UFPE.